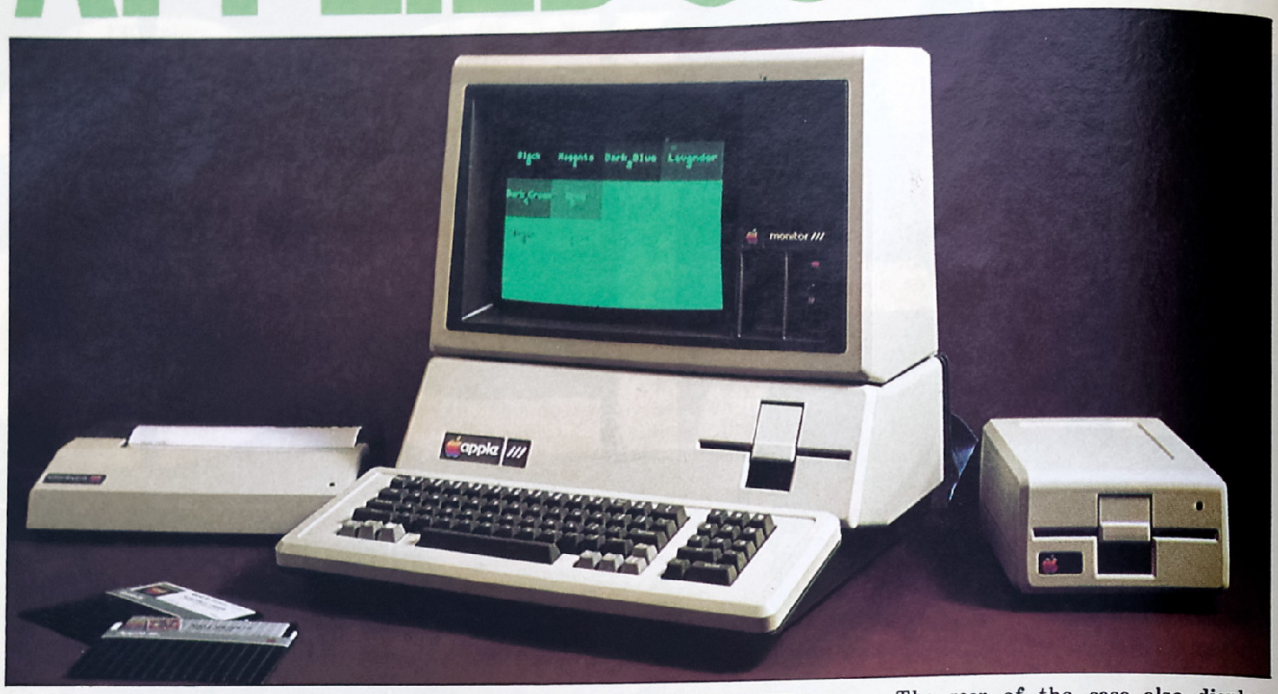
APPLIED SCIENCE



BENCHTEST
PERSONAL COMPUTER

Dick Pountain takes a bite at

APPLE III

Apple II was a very hard act to follow. It may not be the biggest-selling micro (the ZX81 has pipped that title), but it's certainly the best-known and possibly the best-loved micro so far. It has put its inventors at the head of a multi-million dollar company and made the US stock market take microcomputing very seriously indeed. In fact, the US micro industry is growing more and more to rival the music business for the rapidity with which fortunes are made from smash hit product; surely it will not be long before *InfoWorld* (our industry's *Billboard*) begins to publish a software Hot 100.

Apple's problem, then, was the perennial one of following up a hit single. With so much to live up to, it was almost inevitable that the critics would express disappointment at anything which fell short of the miraculous, and when a premature launch in 1980 revealed a crop of hardware problems the word went out that Apple III is a loser. Having finally got my hands on the relaunched model, I can appreciate the industry's difficulty in enthusing over Apple III but I can't agree with the assessment. The machine's virtues are of a subtle rather than revolutionary kind; the hardware is recognisably a gradual development from Apple II rather than a leap into the 16-bit maelstrom, but the operating environment is a considerable advance on current personal computer standards. It appears to have been influenced more by the latest trends in computer science than by the commercial and business sectors and, like its illustrious predecessor, it has the potential to fill a broad range of applications rather than being a single-purpose machine.

Hardware

The Apple III is based around a single 6502B processor (2 MHz), with Apple-designed support circuitry to allow extended memory addressing. The standard machine comes with 128k of RAM which can be expanded on the main board to 256k without using up any of the expansion slots. Only 4k of ROM are present, containing a bootstrap loader and diagnostic routines, as the system is entirely disk-based.

To a casual glance the computer may look like an integrated unit; in fact, the monitor is separate but cleverly styled into a unified line. The computer unit is a little larger than Apple II, and includes a single 5in disk drive and a non-detachable keyboard. A truly massive alloy casting provides the main frame for this unit, with only a top cover and keyboard fascia being fabricated in regulation Apple beige plastic. This mass of metal is not merely to provide structural rigidity — its main function is to act as a heat sink, since no fan is fitted, and its secondary purpose is to shield RF emissions. The rear face has heavy finning cast into it; after half an hour of use the whole body becomes lukewarm but it never exceeds this temperature, even when left on overnight.

The rear of the case also displays seven assorted I/O ports; a disk drive interface, two joystick ports (A and B), an RS232 port, and outputs for monochrome and colour video and audio to an external amplifier. Joystick port A also doubles as an output to the Silen-type printer. The RS232 port has programmable baud rate and handshake protocols.

Access to the internals is easy since the top case removes via two ¼-turn aircraft-type fasteners but, once inside, little is revealed as the electronics are buried at the bottom of a very full enclosure. The main board fills the whole case bottom and the RAM inhabits a piggy-back board under the keyboard section. By virtue of using 64k RAMs, all the 256k of expansion memory can fit on the piggy-back board; access for maintenance is, in fact, performed from underneath the case and Apple issues a strong warning that this is out of bounds to users. The only free space inside the case is devoted to the four Apple II-style 50-pin expansion slots provided for peripheral cards, including the Profile hard disk controller.

The on-board disk drive is a 5in 143k single sided, double density unit like the Apple II units, and up to three external drives may be daisychained via the interface socket. The 5Mb Profile hard disk does not use this socket and four of these could be supported in addition, the only strain being imposed upon your bank account.

Monitor III is a 12in green screen monochrome monitor with a maximum resolution of 560x192 and the ability to display a 16-step grey scale (see photograph) which allows the colour graphics facilities to be used. The unit I tested was made by Hitachi, though the Apple literature refers to it as the Sanyo monitor; presumably Sanyo makes them too. It sits neatly on top of the computer case though it has its own

power cable and on-off switch and connects to the video socket via a coax cable. The display is sharp, steady and legible in most lighting conditions, thanks to a non-glare filter on the screen and a contrast control on the front panel. To obtain a colour display you have to supply your own colour monitor.

The keyboard has 74 keys, 13 on a separate numeric pad, and is a considerable improvement over the Apple II in having four cursor control keys and the reset safely tucked away in a recess in the top edge of the unit. The keys are nicely shaped and have a positive feel, but it is by no means a luxuriously equipped keyboard by current standards, lacking as it does a delete/rubout, clear/home and other editing keys and the now mandatory row of programmable function keys. Instead, Apple has chosen a different route; the whole keyboard is software defined and its layout is stored in a file in the operating system. Two special keys marked with an open and a solid Apple symbol act as modifying keys in addition to SHIFT, CONTROL and ESCape. By preparing alternative layout files, the functions of the various keys can be altered under program control if desired; or, more practically, an azerty layout for continental European users can be provided. In the standard layout all keys auto-repeat at 11 chars/sec when held down; if the 'closed Apple' key is also depressed this speeds up to 33 chars/sec.

A very superior feature of the keyboard is the type-ahead buffer which accepts up to 128 characters when the computer is too busy to process them. This allows you to type at full speed to programs whose slowness would normally force you to wait for a prompt. Type-ahead input is terminated by the first carriage return. My main complaint about the keyboard is that the cursor keys are set lower than the rest and in an 'L' shape at the bottom right hand corner, which makes the backspace unreachable when touch-typing.

One curious, if trivial, fact emerged during the test. Apple III is a very quiet computer since it has no fan and, during the still of the night, I noticed that it makes a noise while it is processing! It emits a very faint buzz *only* while a program is running; it starts with RUN and stops when the prompt returns to the screen. I like to imagine that it is the sound of bits being crunched, but no one so far as satisfactorily explained what device is responsible.

Software

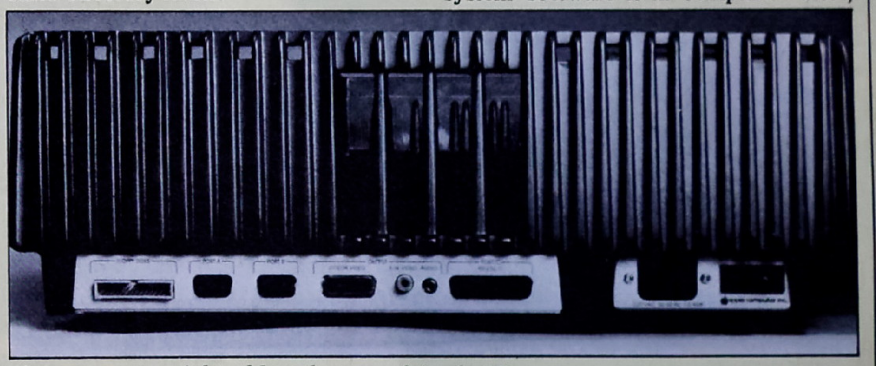
The story of Apple III is very much the story of its operating system, SOS (for Sophisticated Operating System, not Save Our Souls). Rather than follow the herd and opt for CP/M on its second machine, Apple decided to write its own system and I suspect it is from this that much of the adverse reaction has arisen. (Incidentally a Z80 Softcard is planned by Microsoft to allow CP/M to be run.) Leaving aside the commercial wisdom of the decision, I can report that SOS is much more rational and graceful in operation than CP/M, though it is no less intimidating to an end-user.

The design of SOS is heavily influenced by UCSD Pascal and by Unix; the philosophy of SOS is to provide a completely uniform environment for all the languages and all the devices supported by Apple III.

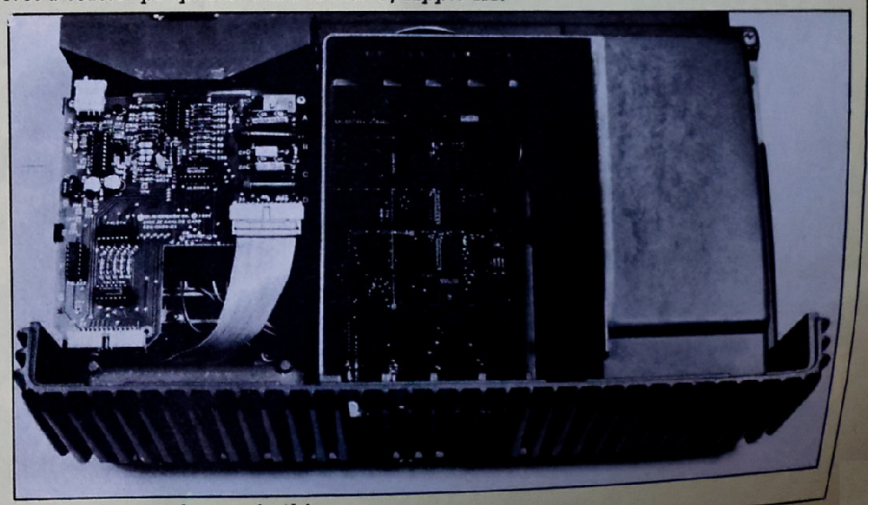
The legacy of Unix is the concept of hierarchical directories. Files may be organised under sub-, sub-sub-directories and so on to any depth necessary. For instance, a customer Smith's file may sit on a volume called ACCOUNTS under a main directory heading PURCHASE in a sub-directory called CARPETS. To inspect Smith's file you would specify the 'pathname' /ACCOUNTS/PURCHASE/CARPETS/SMITH which leads you down a unique path in the branching tree of files. A volume may have 51 main directory entries and 1663 files per sub-directory, which ought to be enough. At first sight it appears as if a lot of typing is necessary (file and directory names can have up to 15 characters). However, a facility called the PREFIX can save a lot of effort. By setting the PREFIX to /ACCOUNTS/PURCHASE/CARPETS/ you can access any customer file in this directory by merely typing the name, eg, JONES. On the other hand, if you set the PREFIX to /ACCOUNTS/PURCHASE/ then you could look at customers of other goods by typing FURNITURE/HIGGINS or DRAPERY/WILSON. In other words, you can focus in on an area of your files where you are currently working. Of course, any file in another part of the system can be accessed at any time by typing its full pathname. This arrangement means that, to SOS, it is of no consequence whether you are using a 140k floppy or a 5Mb winchester — all SOS wants to know is the pathname. The advantage of this system will only really emerge on large storage devices like Profile where the file structure just 'grows' into the available space by using extra directory levels.

The legacy of UCSD is that all devices attached to the Apple III are treated as files. SOS recognises 'character' devices such as printers, the keyboard and the VDU and 'block' devices which are floppy or hard disk units. Block devices have a volume name as well as a device name; for example, the built-in floppy drive is device .D1 but the volume name attaches to the disk in the drive, eg, ACCOUNTS. When specifying the pathname of a file, either the device or the volume name may begin it. Unlike CP/M, SOS automatically logs the volume name of a disk in a drive and so it will search all the drives present for a named volume; it can also prompt you to put a named volume into a certain drive. If you move a volume to another drive, SOS still finds it, which removes the source of a lot of annoying BDOS ERRORS and reboots under CP/M. While on this subject, SOS has over 60 error messages which are in comprehensible English; for example, 'Invalid Pathname' or 'Interpreter File not found' or 'Disk drive not present/not configured'. More important still, SOS invariably fails gracefully when these errors are encountered and allows you to try again in a correct manner without a reboot. The boot ROM performs RAM, ROM and various other hardware tests on power-up and can display certain diagnostic messages in the event of failure.

SOS is composed of three modules, all of which must be present on a bootable disk. The system is always booted from .D1, the built-in drive. SOS.KERNEL contains the nuts and bolts interface to the computer, including the management of the paged memory. SOS.INTERP contains a Pascal p-code interpreter (most of the system software is in compiled Pascal)



Not a toaster peripheral but the rear of Apple III.



Not much wasted space in this case.

A void has been filled!

MatheMagic®

The unique software product that harnesses the power of your Microcomputer to perform simple arithmetic to sophisticated mathematics.

MATHEMAGIC, the friendly, menu driven system that anyone can easily master, is the indispensable tool for business people, educators, students, engineers, scientists and the practical home owner – anyone, in fact, who needs to calculate anything from the simplest arithmetic to the most sophisticated mathematics. In minutes, your micro-computer responds precisely to the task you need to accomplish. MATHEMAGIC is not a "Spreadsheet". Existing products can manipulate columns of numbers but MATHEMAGIC has the broader, almost limitless capability, to perform a whole universe of arithmetic/ mathematical applications.

ism
International Software Marketing,
Hayden House, 5-6 Millmead,
Guildford, Surrey, GU2 5BE.
Tel: 0483-503603

Suite 421, University Building
120 E. Washington Street
Syracuse, New York 13202

Dealer enquiries invited

MATHEMAGIC is available for Apple II or II + ; available soon for Atari, Models 1, 2 & 3, Z-80 based micros with CP/M 2.2, Commodore Pet, CBM and IBM PC.

MatheMagic

There is nothing comparable . . . . . at any price

APPLE III

plus the appropriate language interpreter (eg, Basic), while SOS.DRIVER contains a selection of device driver programs to drive peripheral devices. This latter module can be configured to the user's needs by one of the utility programs and is roughly equivalent to CP/M's BIOS. These three modules typically occupy over 60k, which isn't disastrous with 128k of RAM to play with as you usually have around 50k left for its application programs. However, it reduces the available space on a floppy to less than 80k for a boot disk, which leads to inconveniently large numbers of disks being handled. For serious business use, the Profile would be almost obligatory. Another consequence of the obesity of SOS and the anorexia of Apple floppies is that the System Utilities cannot inhabit a work disk as they can under CP/M. The Utilities disk is filled to within a block of its 140k capacity. Whenever formatting, copying, file management or system configuration is required this disk must be inserted and booted. All the utilities operate through similarly formatted screen menus and I have to admit that they are superbly designed. The top part of the screen contains instructions from SOS plus a menu of options or files to be worked on, one of which is highlighted in reverse field. To choose an option, either type in a single initial letter or use the ↑ cursor key to move the highlight and press return. The bottom of the screen contains your instructions to SOS which initially are set to defaults which SOS thinks are what you most likely wish to do, such as Copy from .D1 to .D2. You can edit these lines to what you want with full use of insert and delete; if you use menu options the names are typed for you by SOS; if you get in a mess you can restore the defaults with a single keystroke.

The main menu offers the options of Device Handling, File Handling or System Configuration. Device Handling leads to a menu offering disk formatting, volume copying (ie, backing-up) and listing devices (and volumes) in the system which in turn lead to further menus. At any point you can go deeper by pressing return to accept the default/edited line or retreat using ESC to the previous menu level. The Filer offers listing of directories, creating new directories, copying, renaming and deleting files, file protection status and setting the time and date (all SOS files are automatically date-stamped with a creation date and date-last-modified).

System Configuration consists of choosing which device drivers you need on a particular boot disk and forming a new SOS.DRIVER file for it. A new SOS system is then generated and put onto the disk. In addition, certain device parameters can be set up, such as for a new type of printer. The standard drivers are .CONSOLE, which is always required, .PRINTER or .SILENTYPE, .GRAFIX, to allow use of the hi-res graphics, .RS232 and .AUDIO for the sound generating facility. The four floppy disk drivers are built into SOS.KERNEL. A driver may be made inactive, in which case it is still there on a disk but is not loaded on bootup; you could have two drivers for different printers and activate the appropriate one when required. Drivers for new peripherals will be supplied on a disk when you buy the device.

The level of help and error trapping throughout these utilities makes them fast and enjoyable to use and puts to shame any other single user personal computer operating system I've had dealings with. A quite inexperienced user can set up a new system configuration with ease, given the driver programs.

SOS may be interrogated from within the various languages by system calls to discover the status of devices or to change parameters like the size of the type-ahead buffer or the screen display width or scrolling area. Such calls are usually optional, however, as the job can be done through invokable modules. A nice touch is that a formatted disk can be scanned and the numbers of bad sectors listed, which allows uncorrupted files to be moved to a good disk.

The principal disadvantages I found in SOS are that a frightening number of different file types are supported (14, if you include UNKNWN!) and that, despite the provision of PREFIX, it is still more wordy than most rivals and gives you lots of typing practice. For instance, to list a program under Basic on the printer involves OPEN#1 AS OUTPUT .SILENTYPE: OUTPUT#1: LIST by dint of its regarding the printer as a file.

Basic

The Basic provided for Apple III is *not* Applesoft but a new Business Basic (though Applesoft programs can be run in emulation mode: see later). This is quite like Microsoft V 5.0 but with a number of interesting extra features.

To my great disappointment, the editing facilities are little better than those on Apple II: to edit a program line you have to press ESC to enter edit mode, move cursor to the line, leave edit mode, retype the line using → and then overtype the alterations. There is no insert facility. The manual suggests that you may wish to use the Pascal text editor to write Basic programs and I fully concur. No AUTO line numbering is provided but DELete is and RENUMbering is available through one of the Invokable Modules (see later). Tracing is possible via TRACE/NOTRACE.

Control structures are normal, with the addition of a (single line) IF. . . THEN. . . ELSE but not WHILE. . . WEND, and the very useful ON KBD GOSUB or GOTO, which allows branching on any key being pressed. KBD is a system variable which holds the ASCII value of the last key pressed. This is a deferred command — ie, the program continues executing past the line containing it until a key is pressed; consequently, it must be reset before it can be used again. Similar structures are ON ERROR. . . for error trapping and ON EOF. . . for reading files.

Variable names can have up to 64 characters, all significant, and four types are supported. Reals have 6-digit mantissas and a range of $10^{\pm 38}$ stored in five bytes, Integers have the suffix % and a range ±32768 stored in three bytes, while Long Integers hold 19 digits, the suffix & and occupy nine bytes. The significance of long integers (a borrowing from UCSD Pascal) for business programs is that all calculations can be done in pence using them and all rounding errors are thus avoided.

Strings have the usual $ suffix and the string functions INSTR$ and SUB$ for substring searching are included as well as the more usual ones. Arrays of any number of dimensions are allowed.

Formatting of displayed output is performed by an extremely powerful PRINT USING statement which uses the additional statement IMAGE to hold a string of 'specs', eg:

```
100 IMAGE 6A,5#,#.6Z4E
200 PRINT USING 100 . . .
```

The various letters and symbols in the IMAGE line specify left or right justification, centering, number of characters in a field, spaces, carriage returns, leading zeros, asterisk fill, and even the insertion of literals into the formatted output. For purely numeric output, fixed point, scientific and engineering formats can be specified as well.

CATALOG [pathname] will list directories or sub-directories and files without having to enter SOS. Both random and sequential files are supported using OPEN#, CLOSE#, CREATE, INPUT# and PRINT#; a nice facility is TYP which returns the type of the next data item to be read.

Interfacing with machine language is rather unorthodox in Business Basic as there are no PEEK, POKE, CALL or USR facilities. Instead, external functions and procedures are supplied as 'Invokable Modules' which are stored on disk and loaded by the statement INVOKE [name]. Once loaded, the various functions in the module can be used from a Basic program by PERFORM [procedure name] or EXFN [function name]. The hi-res graphics reside in such a module called BGRAF.INV as does the renumbering program RENUMBER.INV. Other modules include READCRT, which reads the contents of a screen location, DOWNLOAD, which loads a new character set, and REQUEST and VOLUMES, which list devices statuses and volumes present. Included in REQUEST are procedures FILREAD and FILWRITE, which read or write a specified number of bytes directly from a file. When invoked in a program, the module procedures usually require a list of parameters which can be constants, variables or addresses of variables. The first time I invoked RENUMBER and tried to use it I discovered that it took about eight parameters of different types and hastily concluded that it was a real pain to use. Eventually I discovered, however, that each of the supplied modules has two supplementary files, eg READCRT and READCRT.DOC; the DOC file being a text explaining how to use the package and the other being a version in Basic which can be run from the keyboard rather than invoked in a program, and which is invariably menu-driven. None of this is explained in the manuals, probably because the modules were introduced late and are self-documenting once you know about them!

All of the .INV modules have the filetype PASCOD, which indicates that

APPLE III

they are in p-code, although the manuals imply that they are in Assembler.

Pascal

I really should have discussed the Pascal before the Basic; only tradition prevented me. In many ways, Apple III is a Pascal machine in that the operating system uses many Pascal-like features and is partly written in Pascal, and even the Basic has many features imported from Pascal. In particular, the system of Invokable Modules means that many facilities such as the graphics are shared by both languages.

The version of Pascal adopted is UCSD Pascal 2.1 and, not presuming to be an expert in the system, I can do little more than describe its non-standard features. This was, in fact, my first encounter with UCSD and I was surprised to find the environment much less severe than I had feared; not much harder in fact than an interpreted language. The system has three main levels, the command level, the editor and the filer. At each level the top line of the screen displays a menu of available commands, only the first letter of which need be typed. From command menu you get into the editor, which is very easy to use. It is a screen-oriented affair with insert and delete and the capability to copy large chunks of text via a buffer which holds deleted text. It also assists with the indenting of program text by remembering the margin spacing. Once your source program is ready it is stored in your workfile, you return to command level and hit C for compile, whereupon your program is compiled to p-code. When you run the p-code file it is interpreted into 6502 machine code. The Compiler feeds you with information on the size of the compiled code and any errors encountered. The filer offers the same file management facilities as the SOS Filer, which is in fact just a tarted-up version of it. Also included in the system is a macro assembler and a library of pre-compiled units which can be linked into or called from programs. The system is supplied on three disks, but creating a work disk with a sensible amount of space requires juggling the components around and creating a two-stage boot system with SOS on a separate disk.

Apple III Pascal has some extensions to UCSD 2.1; in particular the floating point maths is upgraded to meet the IEEE standard. Unfortunately, I was unable to run the *PCW* Benchmark 'maths' because the SIN and EXP functions are not in the main language but in library units called REALMODES and TRANSCEND. After adding a USES statement to link these modules, the compiler kept insisting that they were not in the library and, as performing a library map showed no sign of them, I gave up.

Other additions to the standard include the datatypes Wordstream and Bytestream, which effectively allow the creation of arrays whose size is defined at runtime, Treesearch, which performs a fast search for an eight-character name in an alphabetically ordered binary tree and a group of byte-level routines to fill, move and search blocks of memory. Also added is an 'Otherwise' clause for the Case statement.

Graphics

The simplest of Apple III's graphic modes is the text mode, which is the normal operating mode. Even in this mode there are a number of options. Four different character sets are stored on disk and can be loaded by the module DOWNLOAD.INV. Two of the fonts, APPLE (the style used for the company logo) and BYTE are so fancy that they render text virtually unreadable, but ROMAN, which has square serifs, is rather pleasant.

Three different text modes are available, namely 0, which is 40 character/line, black and white; 1, which is 40 char colour; and 2, which is 80 char black and white and is normally set on boot-up. These can be selected from Basic or by system calls. A scrolling area smaller than the screen can be set, cleared or reset either from Basic or by sending an ESC code to the console driver.

The high resolution graphics are obtained through the invokable module BGRAF.INV (PGRAF Unit in Pascal), which contains a variety of routines to draw lines, set colours, fill areas and define shapes. To use the graphics the driver must be first opened with OPEN#1, .GRAFIX, then BGRAF.INV is INVOked and then the various instructions can be PERFORMed.

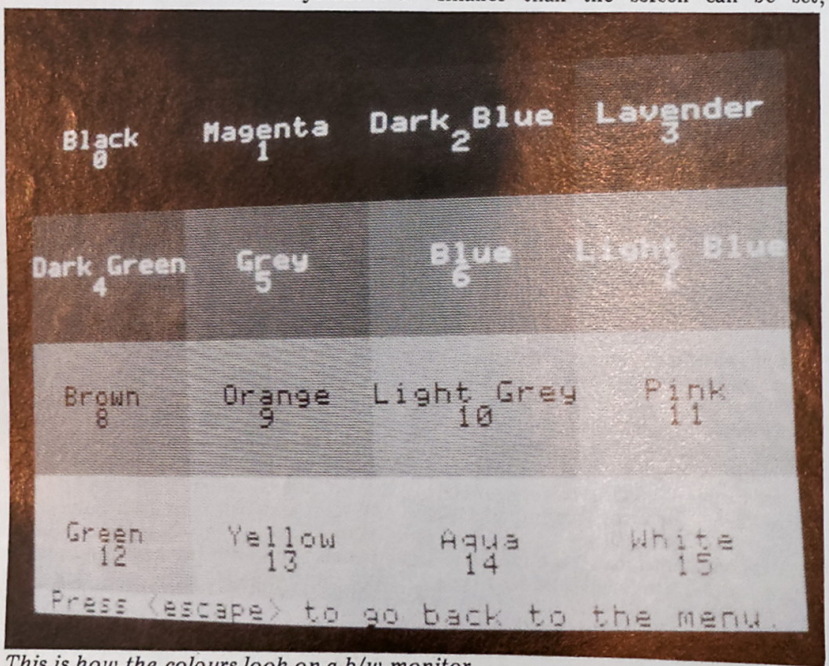
Four graphics modes are available: 280x192 in restricted colour or monochrome; 560x192 in black and white only; and 140x192 in 16 colours. On the Monitor III the colour modes show in shades of grey. Each mode uses two buffers so that two separate screens can be drawn and displayed alternately, allowing animation effects. The graphics routines work fast enough to produce reasonable effects, as witnessed by the galloping horses in the demo package. GRAFIXMODE selects a mode and buffer to display, while GRAFIXON switches from text to graphics mode and displays the buffer. TEXT returns to text mode. Text can be freely mixed with graphics, using either the current system character set or a user-defined one.

Two sorts of plotting command are available, those which use absolute screen coordinates (DOTAT, LINETO, MOVETO) and those which plot relative to the last point plotted (DOTREL, LINEREL, MOVEREL). The latter would make implementation of Turtle graphics a snap.

A very sophisticated system exists for colour control. As well as defining the background colour and 'pencolor' (ie, the colour plotted), it is possible to define relationships between colours which overlay one another. The command SETCTAB (%11,%8,%9) means 'wherever pink is placed over brown make the result show as orange', which is invaluable for drawing maps or circuit layouts, as well as being a lot of fun. Also, by using the so-called 'transfer options', a bewildering variety of Boolean operations can be performed between back and foreground colours, eg [NOT foreground] XOR background. If you can fathom them, these allow figures to be transparent or opaque and give many other effects.

The shape-defining routine is surprisingly crude by comparison; it sets up a bit map of the shape represented in a (decimal) integer array and allows it to be placed anywhere on the screen (but not scaled or rotated). It is a real pig to use as the principal command, DRAWIMAGE, takes six parameters, one of them an array address. A related routine NEWFONT is used to define character sets; I wish you the best of luck but I'll stick to Sirius's EDOT, thank you.

Finally, GLOAD and GSAVE allow you to preserve screen images as disk files (type .FOTO) and reshow them later.



This is how the colours look on a b/w monitor.

Apple II emulation

Apple III can be persuaded to think it's an Apple II to run the vast number of programs which have been produced over the years for the earlier machine.

On booting the emulation disk, a menu is displayed which allows certain

limited configuration options to be chosen. The default is a 48k Apple II Plus with 16-sector disk, Applesoft Basic and a serial card, but you could choose Integer Basic and set various printer parameters to suit the program you wish to run.

There are some limitations to the emulation: no programs which require support from a language card can be run and hi-res colour graphics cannot be displayed on an RGB monitor. Pascal programs written for Apple II can be recompiled on the III as it can read the source disks. Another problem is that the keyboards produce some different codes; a table in the manual shows where unexpected results may occur.

This feature should be very attractive to Apple II owners who wish to upgrade and already have an extensive software library, and it makes a lot of commercial sense in the US where an ocean of Apple II software exists.

Documentation

To call this documentation comprehensive would be an understatement. I had no fewer than eight manuals: Owner's Guide, Standard Device Drivers Manual, Business Basic vols I & II, Pascal Introduction Filer and Editor, Pascal Programmer's Manual vols I & II, and Pascal Program Preparation Tools. All are well-written in a friendly but authoritative style and are nicely produced in spiral binders. Just about everything you want to know is in there somewhere. The only exception I know of is that the Basic invokable modules apart from BGRAF are not covered. However, it is often the case (as with the Apple II manuals) that you don't know which volume to look in. . . Each book has a good index but an overall index would be a great help with a library of this size in which some subjects like SOS are spread among different volumes.

There is, in addition, an SOS System Reference Manual, presumably for professional programmers, which I didn't have for the test. Given time to become familiar with it, this documentation is as good as any on the market.

Expansion and potential use

The system as tested could have been expanded to 256k RAM, a Profile 5Mb hard disk (and two more outboard floppies) and a Qume daisywheel printer could be substituted for the Silentype for letter-quality work.

A package called Access III allows communication between Apples via the RS232 or use as an intelligent terminal for timesharing and remote database applications.

Word processing is catered for by a completely revamped version of Applewriter which is far superior to the original (chorus of 'it would have to be!'). As well as supporting an 80-column screen and upper and lower case, it has several of the best features of the Pascal editor, including the delete buffer, which allows you to change your mind if you make an incorrect deletion and also to move small chunks of text without the bother of block markers. A Mail List Manager package makes this into a competitive WP system at last.

Technical specifications

Processor:	6502B (2MHz)
Memory:	128k RAM, 4k ROM
Disks:	Twin 143k 5¼in single-sided, double density
Display:	12in green monitor, 560x192 max graphics resolution, 80 or 40 column text. Comp video & RGB outputs
Keyboard:	74-key ASCII with separate numeric keypad
Audio:	2in speaker with 6-bit DAC
I/O:	RS232, twin joystick, Silentype printer
System software:	Apple SOS
Languages:	Basic, UCSD Pascal, Assembler

Visicalc III is an enhanced version of the famous package, with more extensive editing and conditional branch abilities; it can be used in conjunction with a business graphics package which produces bar and pie charts and performs curve fitting and other plots using Visicalc files if desired.

Other languages promised soon are Cis-Cobol, Trans-Forth and the ALD Assembler, while Fortran must follow fairly soon.

As to the question of who will use Apple III, it should appeal to scientists and engineers for its powerful graphics and the tremendous maths capability of UCSD Pascal. The IEEE floating point includes Affine and Projective modes which can handle arithmetic infinities, as well as improved precision and error checking. A Universal Parallel card is available for instrument interfacing, though I can't say whether this is IEEE – 488 compatible.

Forward-looking programmers ought to love it as the SOS/Pascal environment is very good to work in. This bodes well for a future supply of high quality software once the initial bad press has worn off and sales pick up.

In business, the III would make an excellent management tool for planning and forecasting, the same role which the Hewlett-Packard 125 is aimed at; the hard disk and communications would be a bonus here. It's not as certain how useful it will be to the small business, as it offers very little advantage over a CP/M machine. The pool of ready-made business software is not so large, the floppy disk capacity is small and the hard disk is expensive. Also, although I find SOS superior to CP/M, it is every bit as frightening to the inexperienced end-user – if rather more forgiving.

Apple III would make a very rewarding machine for experienced hobbyists but I fear that, at least in the impoverished UK, it is just too expensive.

Prices

Apple III with Monitor III, SOS, Business Basic, Apple II Emulation and Manuals	£2545
Additional floppy drive	£385
Silentype printer	£222
Qume Sprint daisywheel printer	£1640
Profile 5Mb winchester drive	£2256
Pascal	£150
Visicalc	£150
Universal parallel interface	£135

All prices exclude VAT; all products except Qume include one-year warranty.

Conclusions

In a recent interview Steven Jobs, one of the Applefathers, expressed the opinion that people who are chasing 16 bits and more memory are in the wrong race; that software design is the key to the future. Certainly Apple III embodies this philosophy in that its processor is a second-generation workhorse and there is little that is startling (though much that is neat) in the hardware design. The voice of the computer scientist has been heard above that of the engineer in the development of the machine. The software environment around the III is more rational and sophisticated than that of its direct competitors and is also flexible enough to accommodate a lot of future hardware developments while maintaining program portability.

Whether or not it is a better buy than Sirius or IBM depends on how well the programmers make use of this sophistication, as neither CP/M nor SOS are really fit to be put in front of the end user. The truly friendly operating system is still in the future; it may be in Apple's future with Lisa.

My thanks to Digitus for the loan of the test machine and to Lasky's for the Pascal system.

END

Basic Benchmarks

BM1	1.7
BM2	7.2
BM3	13.5
BM4	14.5
BM5	16.0
BM6	27.0
BM7	42.5
BM8	7.5

Pascal Benchmarks

magnifier	4.5
forloop	53.0
whileloop	50.0
repeatloop	44.5
literalassign	63.5
memoryaccess	65.0
realarithmetic	70.5
realalgebra	61.5
equalif	82.5
unequalif	81.0
vector	144.5
noparameters	39.0
value	42.5
reference	43.5
maths	—